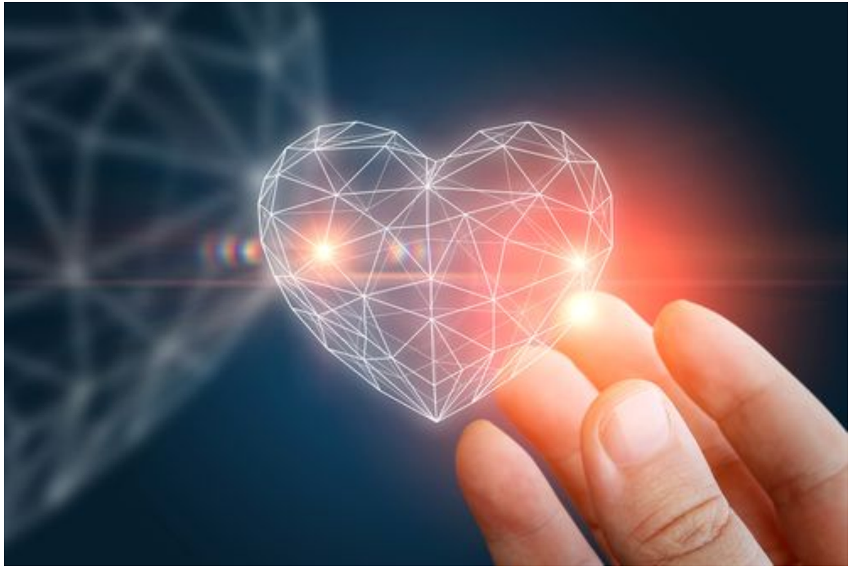
Capstone Project - 3

Cardiovascular Risk Prediction

Let's get the Chronic Heart Diseased patients:

1. Defining Problem Statement
2. Exploratory Data Analysis and Feature Selection
3. Feature Selection
4. Preparing dataset for modelling
5. Applying Model
6. Model Validation and Selection



Cardiovascular Risk Prediction

Cardiovascular disease (CVD) remains the most important cause of morbidity and mortality worldwide.1 For prevention of CVD, cardiovascular risk management is advocated in international guidelines.2 3 Many cohort studies and randomised controlled clinical trials (RCTs) have demonstrated the benefits of risk factor management, including smoking cessation, lipid lowering, blood pressure lowering, antithrombotic therapy, glucose lowering and more recently, anti-inflammatory therapies, on CVD risk.4–9 Besides these interventions, healthy lifestyle behaviour should always be promoted at individual *and* population level. With this growing plethora of choices in cardiovascular prevention, it can be difficult for both healthcare professional and patient to make the most appropriate treatment decisions for each individual person.

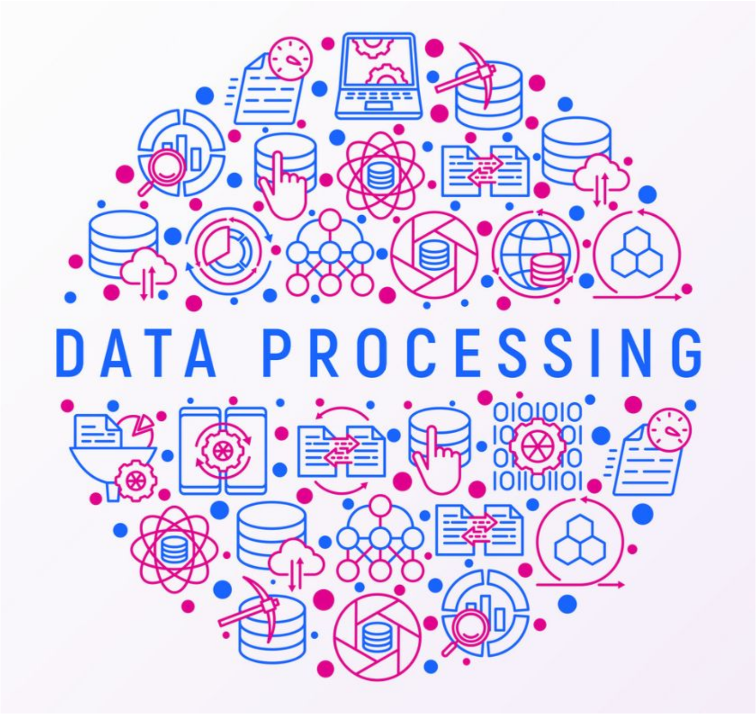
Identifying those patients who will benefit most from risk factor treatment is pivotal in the global CVD prevention effort. Risk stratification is a cornerstone in international CVD prevention guidelines, aiming to identify those at highest risk of future CVD in order to most effectively apply preventive strategies. Risk assessment using risk prediction tools can thus play a highly important part in global CVD prevention efforts in choosing the right treatment and the right treatment goals, for the right patient. This narrative review aims to guide clinicians in using risk stratification tools as decision support tool in CVD prevention.

Data Processing

- **Data Preprocessing**: Deletion of NaN values and replacing it with the respective values to process the data machine readable for ML and DL purposes.

- **EDA**: Exploratory Data Analysis is done on the dataset to get inference from the data and to see the visible trends.

- **Create a model**: Experimenting with different models to get the best possible F1 Score which determines the ability of the model to classify with high accuracy.



Exploratory Data Analysis

In the **Exploratory Data Analysis(EDA)** part, the data is correlated and the trends in the data are discussed. The statistics obtained are as follows:

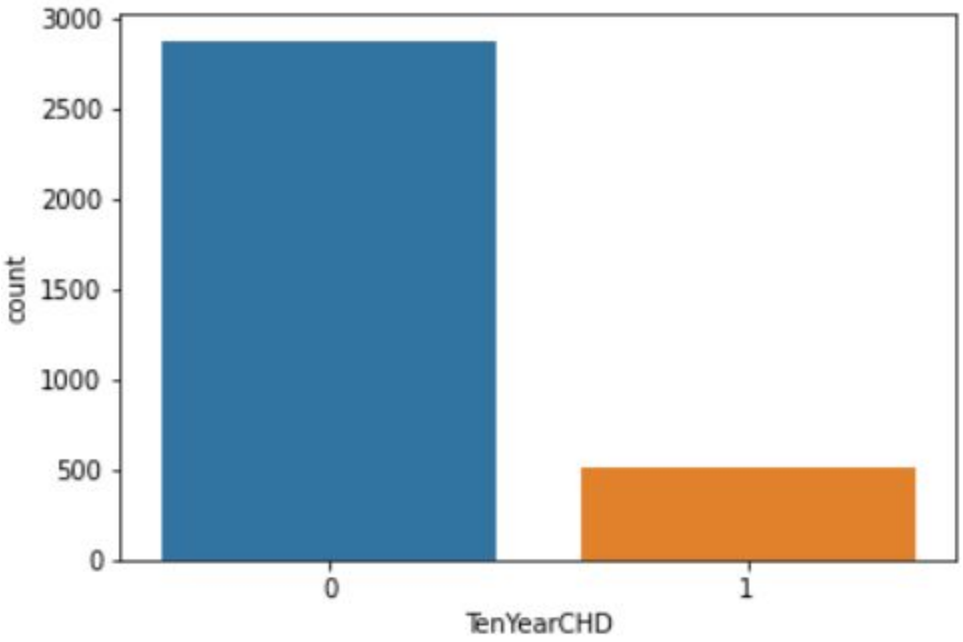
- Checking the distribution
- Treating the null values
- Plotting the variables
- Getting dummies for the categorical variables

Data Summary

Cardiovascular Risk Prediction Dataset:

#	Column	Non-Null Count	Dtype
---	------	--------------	-----
0	id	3390 non-null	int64
1	age	3390 non-null	int64
2	education	3390 non-null	float64
3	sex	3390 non-null	object
4	is_smoking	3390 non-null	object
5	cigsPerDay	3390 non-null	float64
6	BPMeds	3390 non-null	float64
7	prevalentStroke	3390 non-null	int64
8	prevalentHyp	3390 non-null	int64
9	diabetes	3390 non-null	int64
10	totChol	3390 non-null	float64
11	sysBP	3390 non-null	float64
12	diaBP	3390 non-null	float64
13	BMI	3390 non-null	float64
14	heartRate	3390 non-null	float64
15	glucose	3390 non-null	float64
16	TenYearCHD	3390 non-null	int64

Exploratory Data Analysis



Count Plot of TenYearCHD

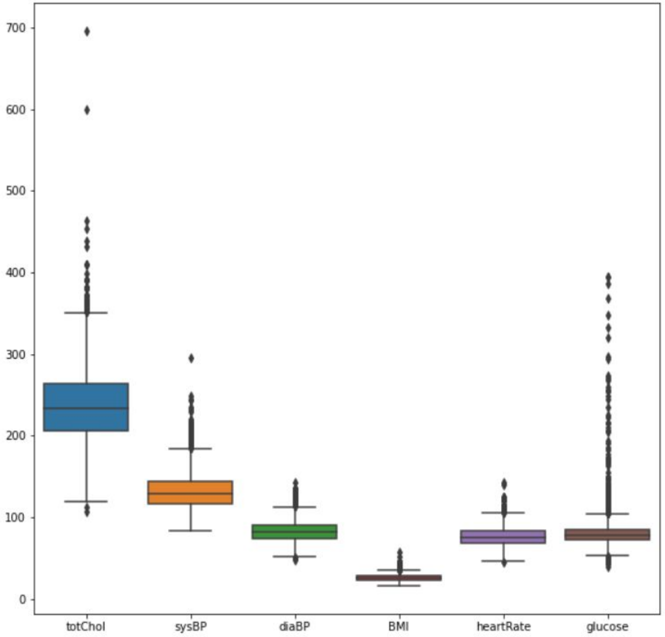
Exploratory Data Analysis

Dummy Variables

sex_F	sex_M	is_smoking_NO	is_smoking_YES
1	0	0	1
0	1	1	0
1	0	0	1
0	1	0	1
1	0	0	1
...	...	...	...
1	0	1	0
1	0	1	0
0	1	0	1
0	1	1	0
1	0	1	0

Exploratory Data Analysis

Box Plot of numerical variables:



Logistic Regression Model

LR Metrics:

```
lr_bayes.best_estimator_
```

```
LogisticRegression(C=2.6681629323335714, class_weight=None, dual=False,
                   fit_intercept=True, intercept_scaling=1, l1_ratio=None,
                   max_iter=100, multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

```
print('Train ROC-AUC score : ', lr_bayes.best_estimator_.score(X_train,y_train))
print('Test ROC-AUC score : ', lr_bayes.best_estimator_.score(X_test,y_test))
```

```
Train ROC-AUC score :  0.6743862899490505
Test ROC-AUC score :  0.6743055555555556
```

Support Vector Machine Model

SVC Metrics:

```
svc_bayes.best_estimator_
```

```
SVC(C=1000, break_ties=False, cache_size=200, class_weight=None, coef0=0.0,
    decision_function_shape='ovr', degree=3, gamma=1, kernel='rbf', max_iter=-1,
    probability=False, random_state=None, shrinking=True, tol=0.001,
    verbose=False)
```

```
print('Train ROC-AUC score : ', svc_bayes.best_estimator_.score(X_train,y_train))
print('Test ROC-AUC score : ', svc_bayes.best_estimator_.score(X_test,y_test))
```

```
Train ROC-AUC score :  0.9418712366836498
Test ROC-AUC score :  0.8416666666666667
```

Decision Tree Classifier

Decision Tree Model Metrics:

```
dt_bayes.best_params_
```

```
OrderedDict([('max_depth', 8),
             ('min_samples_leaf', 10),
             ('min_samples_split', 50)])
```

```
print('Train ROC-AUC score : ', dt_bayes.best_estimator_.score(X_train,y_train))
print('Test ROC-AUC score : ', dt_bayes.best_estimator_.score(X_test,y_test))
```

```
Train ROC-AUC score :  0.8186660490968041
Test ROC-AUC score :  0.7875
```

Random Forest Classifier

RF Model Metrics:

```
rf_bayes.best_params_
```

```
OrderedDict([('max_depth', 8),
             ('min_samples_leaf', 10),
             ('min_samples_split', 50),
             ('n_estimators', 100)])
```

```
print('Train ROC-AUC score : ', rf_bayes.best_estimator_.score(X_train,y_train))
print('Test ROC-AUC score : ', rf_bayes.best_estimator_.score(X_test,y_test))
```

```
Train ROC-AUC score :  0.8659101435849931
Test ROC-AUC score :  0.8368055555555556
```

Gradient Boosting Machine Classifier

GB Model Metrics:

```
gb_bayes.best_params_
```

```
OrderedDict([('max_depth', 8),
             ('min_samples_leaf', 11),
             ('min_samples_split', 52),
             ('n_estimators', 89)])
```

```
print('Train ROC-AUC score : ', gb_bayes.best_estimator_.score(X_train,y_train))
print('Test ROC-AUC score : ', gb_bayes.best_estimator_.score(X_test,y_test))
```

```
Train ROC-AUC score :  0.9759147753589624
Test ROC-AUC score :  0.9006944444444445
```

XGBoost Classifier

XGBoost Model Metrics:

```
xgb_bayes.best_params_
```

```
OrderedDict([('learning_rate', 0.09776808328011032),
             ('max_depth', 10),
             ('n_estimators', 69)])
```

```
print('Train ROC-AUC score : ', xgb_bayes.best_estimator_.score(X_train,y_train))
print('Test ROC-AUC score : ', xgb_bayes.best_estimator_.score(X_test,y_test))
```

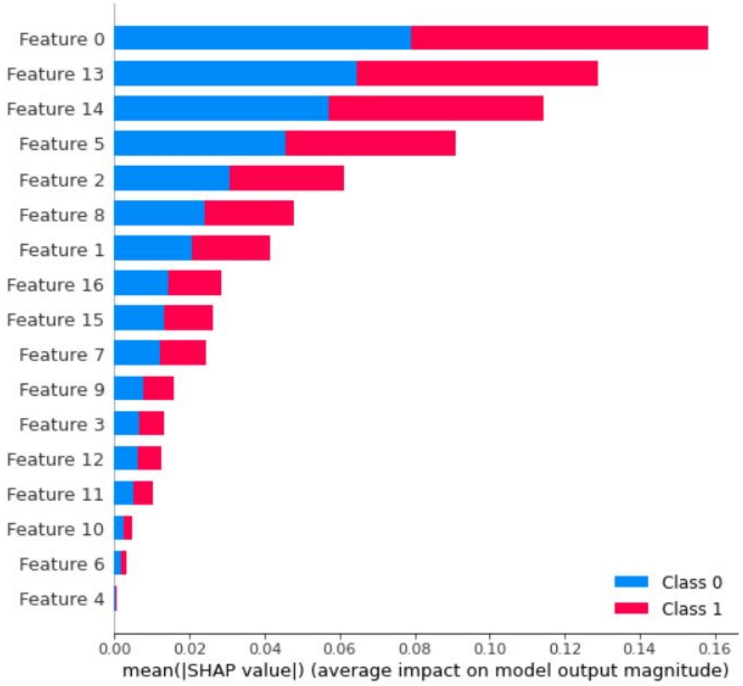
```
Train ROC-AUC score :  0.9925891616489115
Test ROC-AUC score :  0.9020833333333333
```

Let’s Collect the metrics of all of our models:

	Train accuracy	Test accuracy	Train precision	Test precision	Train recall	Test recall	Train f1 score	Test f1 score	Train ROC-AUC	Test ROC-AUC	Model Name
0	0.68	0.67	0.67	0.66	0.69	0.68	0.68	0.67	0.67	0.67	LogisticRegression
1	0.94	0.84	0.95	0.82	0.93	0.86	0.94	0.84	0.94	0.84	SupportVectorClassifier
2	0.81	0.77	0.82	0.78	0.79	0.75	0.81	0.76	0.81	0.77	DecisionTreeClassifier
3	0.96	0.9	0.99	0.93	0.93	0.85	0.96	0.89	0.96	0.89	GradientBoostingClassifier
4	0.86	0.85	0.9	0.88	0.82	0.8	0.86	0.84	0.86	0.85	RandomForestClassifier
5	0.99	0.9	1	0.92	0.99	0.88	0.99	0.9	0.99	0.9	XGBClassifier

Feature Importances(Using Shap Library)

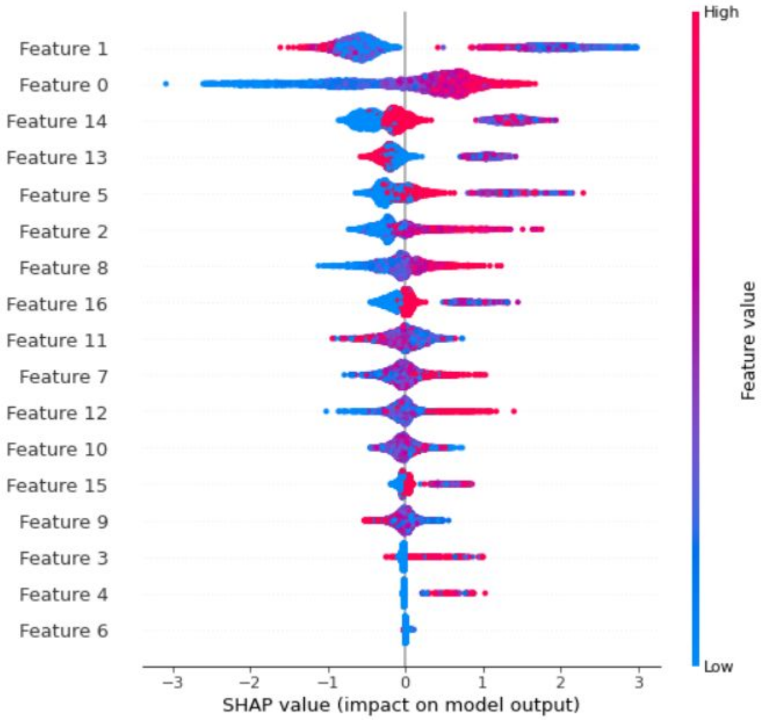
RF Model:




```
Most_important_features = ['age', 'sex_F', 'sex_M', 'prevalentHyp','cigsPerDay']
```

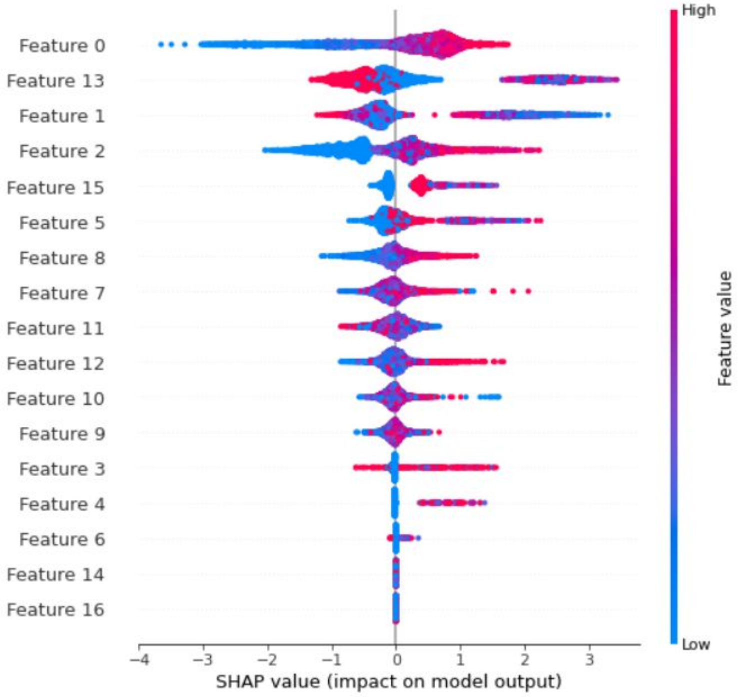
Feature Importances(Using Shap Library)

GBM Model:



Feature Importances(Using Shap Library)

XGBoost Model:



Conclusion

- Upon the tested models XGBoost gives the highest F1 Score which determines the ability of the model to classify the class 0 and 1. XGBoost gives out the score as 0.9 this means that the model can classify at an accuracy of around 90% for both the classes.

- Therefore, the best tested model is the XGBoost model with an accuracy of 90%, Precision of 92%, Recall of 88%, F1 score of 90% and ROC-AUC Score of 90%.

- Therefore, this model can find out whether the person is prone to CVD or not at ~90% accuracy.

	Train accuracy	Test accuracy	Train precision	Test precision	Train recall	Test recall	Train f1 score	Test f1 score	Train ROC-AUC	Test ROC-AUC	Model Name
0	0.68	0.67	0.67	0.66	0.69	0.68	0.68	0.67	0.67	0.67	LogisticRegression
1	0.94	0.84	0.95	0.82	0.93	0.86	0.94	0.84	0.94	0.84	SupportVectorClassifier
2	0.81	0.77	0.82	0.78	0.79	0.75	0.81	0.76	0.81	0.77	DecisionTreeClassifier
3	0.96	0.9	0.99	0.93	0.93	0.85	0.96	0.89	0.96	0.89	GradientBoostingClassifier
4	0.86	0.85	0.9	0.88	0.82	0.8	0.86	0.84	0.86	0.85	RandomForestClassifier
5	0.99	0.9	1	0.92	0.99	0.88	0.99	0.9	0.99	0.9	XGBClassifier

The End